Project Report

1. Introduction

The goal of this project is to classify Iris flowers into one of three species based on their physical measurements. Each flower is described using four numerical features, and the task is to predict the correct species from these values.

Instead of using a standard classification algorithm, this project uses linear regression for classification. The purpose is to understand how a regression model can be trained using gradient descent and then adapted to perform multi-class classification by mapping continuous outputs.

2. Dataset Description

The dataset used in this project is the Iris dataset obtained from the UCI Machine Learning Repository. The dataset consists of 150 samples, each representing a flower from one of three Iris species: Iris-setosa, Iris-versicolor, and Iris-virginica.

Each sample contains four numerical features: sepal length, sepal width, petal length, and petal width. These features describe the physical characteristics of the flowers and are used as inputs to the model.

The original class labels are provided as strings. For applying linear regression, the class labels were mapped to numeric values as follows: Iris-setosa to -1, Iris-versicolor to 0, and Iris-virginica to 1.

The dataset was split into a training set of 120 samples and a testing set of 30 samples. A stratified split was used to ensure that each class was equally represented in both the training and testing datasets.

set of 120 samples and a testing set of 30 samples. A stratified split was used so that each class was equally represented in both sets.

3. Data Preprocessing

Before training the model, several preprocessing steps were applied to the dataset. First, all class labels were converted from strings to numeric values. This conversion was required because linear regression operates on numerical targets.

Next, feature normalization was applied using Min-Max normalization to scale each feature to the range [0, 1]. Normalization improves the stability and convergence speed of gradient descent by ensuring that all features contribute proportionally to the learning process.

$$x' = \frac{(x - x_{\{min\}})}{(x\_\{max\} - x\_\{min\})}$$

4. Methodology Overview

1. The overall methodology followed a structured pipeline:
2. Load and parse the Iris dataset
3. Map class labels to numeric values
4. Perform five-fold stratified cross-validation: for each fold, fit Min-Max normalization on the fold's training data, train the model, and evaluate on the fold's validation data
5. Create a final stratified 120/30 train–test split for final evaluation
6. Fit Min-Max normalization on the final training set and apply it to both training and test sets
7. Train the linear regression model using batch gradient descent on the normalized training set
8. Convert continuous regression outputs into discrete class labels using the nearest-label rule {−1, 0, 1}
9. Evaluate performance using classification accuracy on the test set
10. Test the trained model on manually selected samples and report predicted class names

5. Linear Regression Model

Linear regression was used as the underlying model in this project. The model computes a linear combination of the input features along with a bias term to produce a continuous output.

$$\hat{\boldsymbol{y}} = Xw + b$$

6. Loss Function and Training

The model was trained using batch gradient descent to minimize the Mean Squared Error between predicted outputs and true numeric labels. The loss (cost) function is:

$$J(w, b) = \frac{1}{2m} \sum_{i=1}^{m} (\hat{y}_i - y_i)^2$$

During each iteration of training, the gradients of the loss function with respect to the model parameters were computed manually, and the weights were updated using a fixed learning rate. Training was performed for a fixed number of iterations, and the loss value was recorded at each step to observe convergence.

7. Classification Strategy

Because linear regression produces continuous outputs, an additional decision rule was required to perform classification. Each regression output was mapped to a discrete class label using a nearest-label strategy.

The set of valid class labels was defined as $\{-1, 0, 1\}$. For each prediction, the label with the smallest absolute difference from the regression output was selected as the final class. The numeric labels were then mapped back to their corresponding Iris species names for interpretability.

8. Visualizations

This section contains all figures generated during the project.

Figure 1 shows a scatter plot of petal length versus petal width for the entire dataset. This plot illustrates how the three Iris classes are distributed in feature space and highlights the separability of the classes.

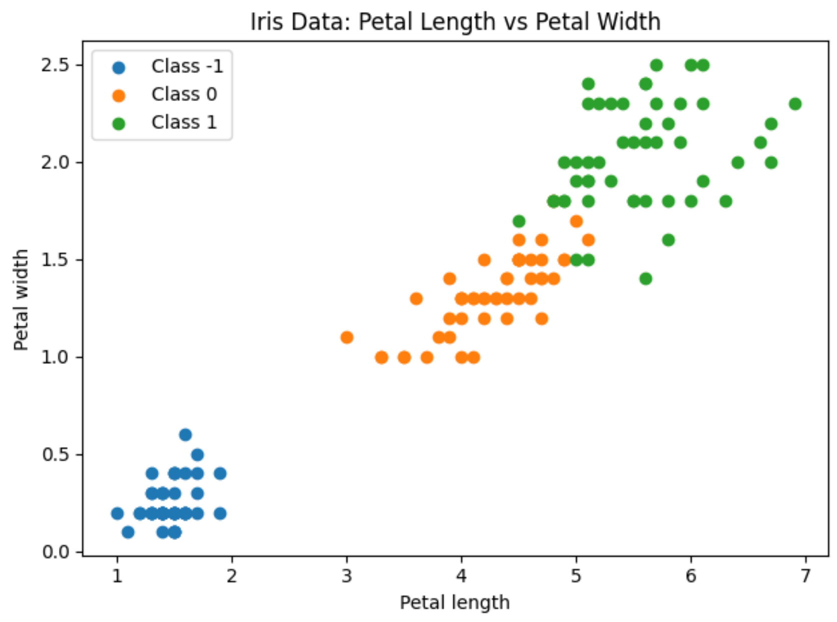


Figure 2 shows a one-dimensional slice of the linear regression model before training. Petal length is varied while the remaining features are held at their mean values. This plot represents the initial, untrained model.

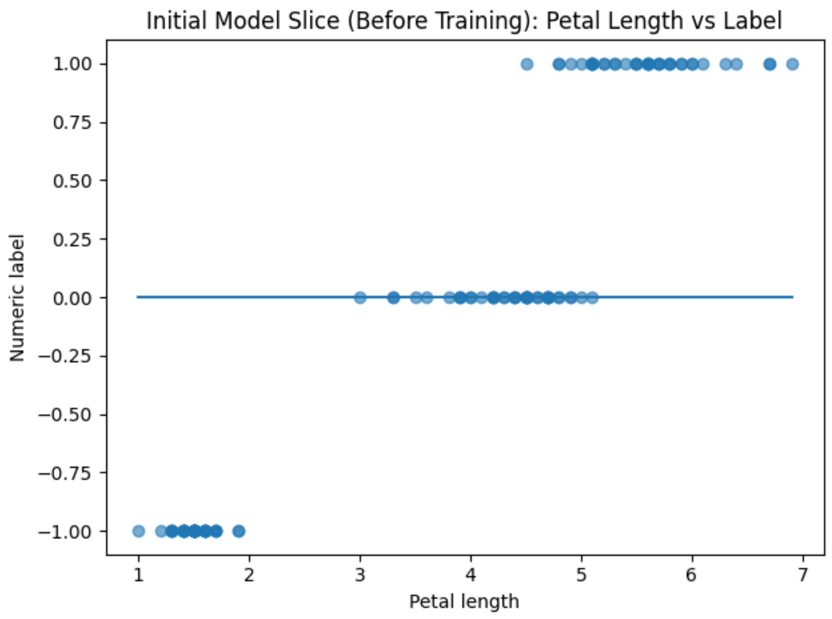


Figure 3 shows the same one-dimensional slice after training using gradient descent. The learned regression line reflects how the model has adapted to fit the data.

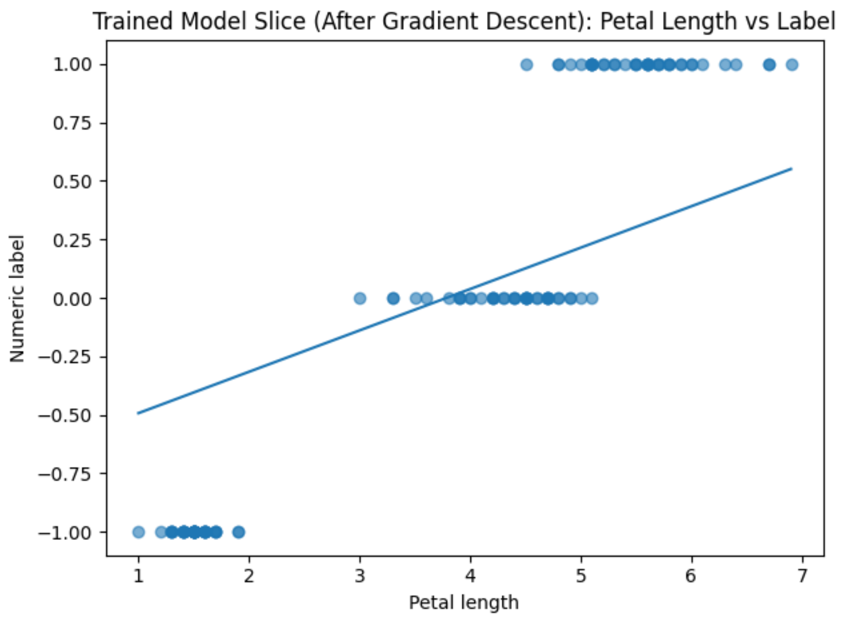
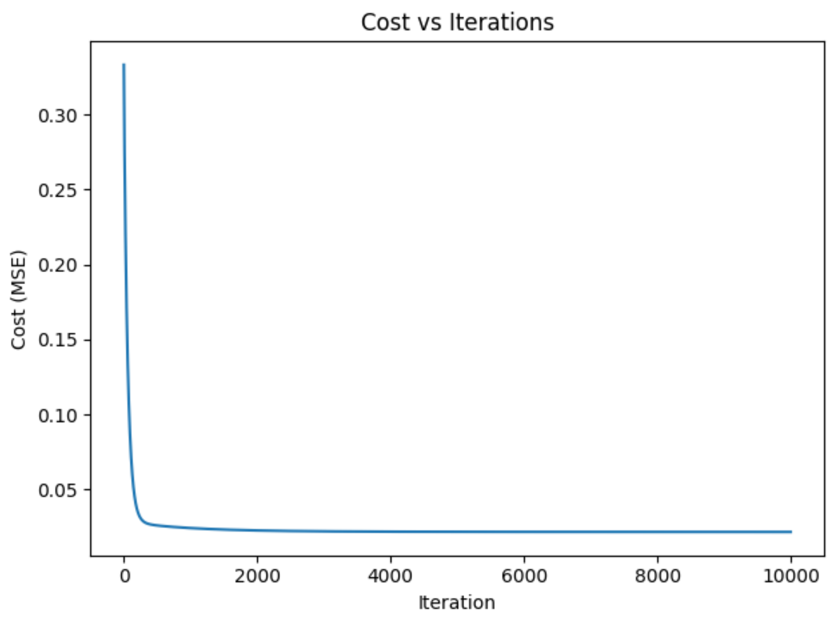
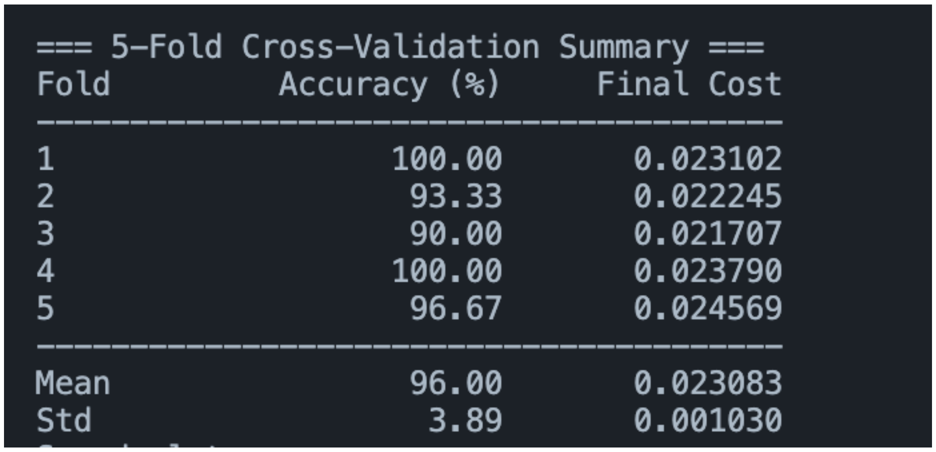


Figure 4 shows the training loss versus the number of iterations. The decreasing curve demonstrates stable convergence of the gradient descent algorithm.



1. Results

Five-fold cross-validation was used to validate the stability of the model. The mean validation accuracy across the five folds was 96.00%**,** with a standard deviation of 3.89%**.**

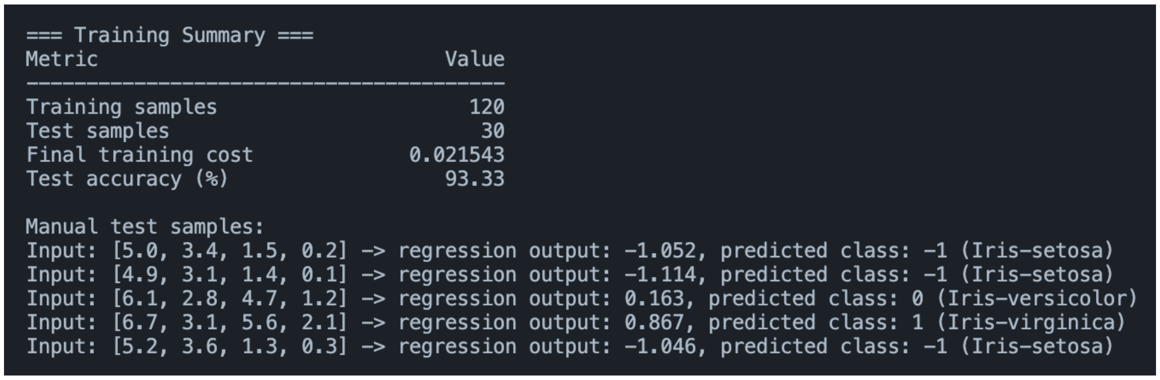


```
=== 5-Fold Cross-Validation Summary ===
Fold          Accuracy (%)     Final Cost
----------------------------------------
1                   100.00       0.023102
2                    93.33       0.022245
3                    90.00       0.021707
4                   100.00       0.023790
5                    96.67       0.024569
----------------------------------------
Mean                 96.00       0.023083
Std                   3.89       0.001030
```

For final evaluation, the dataset was split into 120 training samples and 30 testing samples. The trained model achieved a test accuracy of approximately 93**%** on the unseen test set.

The final training cost achieved was 0.0215, indicating good convergence of the gradient descent algorithm.

In addition to quantitative evaluation, several manually selected test samples were passed through the trained model. Samples resembling Iris-setosa, Iris-versicolor, and Iris-virginica were correctly classified, demonstrating the model's practical effectiveness.



```
=== Training Summary ===
Metric                                 Value
----------------------------------------
Training samples                         120
Test samples                              30
Final training cost                 0.021543
Test accuracy (%)                      93.33

Manual test samples:
Input: [5.0, 3.4, 1.5, 0.2] -> regression output: -1.052, predicted class: -1 (Iris-setosa)
Input: [4.9, 3.1, 1.4, 0.1] -> regression output: -1.114, predicted class: -1 (Iris-setosa)
Input: [6.1, 2.8, 4.7, 1.2] -> regression output: 0.163, predicted class: 0 (Iris-versicolor)
Input: [6.7, 3.1, 5.6, 2.1] -> regression output: 0.867, predicted class: 1 (Iris-virginica)
Input: [5.2, 3.6, 1.3, 0.3] -> regression output: -1.046, predicted class: -1 (Iris-setosa)
```